Algorithmique
TD 6. Semaine du 12/11/2001

Exercice 1 : Diviser pour régner, quelques calculs

1) On considère la suite (u_n) définie par les relations de récurrence $u_1 = 1$ et $u_n = 1 + u_{\lceil n/2 \rceil}$. Montrer que u_n est le nombre de comparaisons dans le cas le pire pour la recherche d'un élément dans un arbre de recherche équilibré de taille n. Montrer, sans calculs, que $u_n = \lfloor \log_2 n \rfloor + 1$.

2) Soit $u_1 = 0$, $u_{2p+1} = 2p+1+u_{p+1}+u_p$ et $u_{2p} = 2p+2u_p$. Montrer que u_n est le nombre de comparaisons en moyenne du tri Quicksort d'un tableau de taille n. On pose $v_k = u_{2^k}$. Montrer que pour tout $k \geq 0$, $v_{k+1} = 2^{k+1}+2v_k$. En déduire par récurrence que $v_k = k2^k$ et donc que si n est une puissance de 2, $u_n = n \log_2 n$.

3) Soit $u_1 = 0$, $u_{2p} = \frac{p(p+1)}{2} + 2u_p$ et $u_{2p+1} = \frac{p(p+1)}{2} + u_p + u_{p+1}$. Montrer que u_n est le nombre d'échanges dans le cas le pire du tri Shell. On se propose de calculer u_{2^k}. On pose pour $k \geq 0$, $v_k = \frac{u_{2^k}}{2^{k-2}}$. Montrer que $v_{k+1} = 2^k + 1 + v_k$ et que $v_0 = 0$. En déduire que $v_k = 2^k - 1 + k$ et que pour $n = 2^k$, $u_n = \frac{n^2}{4} + \frac{n \log_2 n}{4} - \frac{n}{4}$.

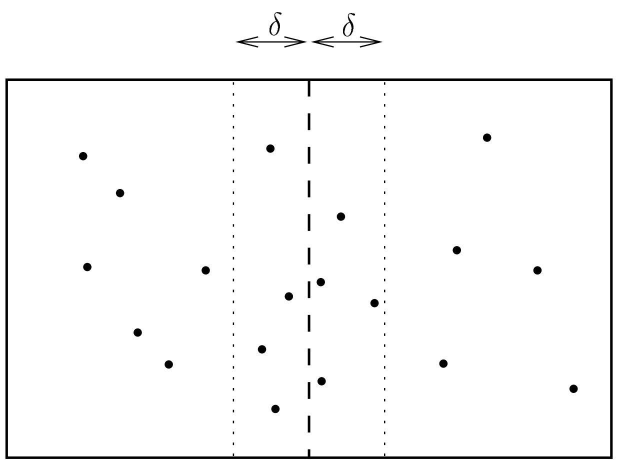
Exercice 2 : Diviser pour régner

On se propose de chercher parmi n éléments d'une liste le plus grand et le plus petit.

1) Trouver un algorithme simple de compléxité en nombre de comparaisons inférieur à $2n - 3$.

2) On utilise la méthode récursive suivante : on divise la liste en deux sous-listes, on calcul le minimum et le maximum de chacune et on compare entre eux les minima et les maxima obtenus.

Écrire l'algorithme correspondant à cette méthode et calculer sa compléxité en nombre de comparaisons. Cet algorithme est-il meilleur que le précédent ?



Exercice 3 : distance minimale dans un nuage de points

Étant donner un tableau de n points du plan on cherche à déterminer la distance minimale entre deux points de ce tableau.

1) Proposer un algorithme en $\Theta(n^2)$ calculs de distance.

2) Montrer que l'algorithme suivant est en $\Theta(n \log n)$:

1. Étant donnée un tableau T de n points :
2. trier `T` par ordre croissant des abscisses;
3. calculer les distances minimales δ_g et δ_d pour les sous-tableaux triés `T[0...n/2]` et `T[n/2...n]`;
4. $\delta = \min(\delta_g, \delta_d)$;
5. construire la bande médiane de largeur 2δ;
6. trier la partie gauche et la partie droite de la bande médiane par ordonnées croissantes;
7. calculer pour chaque point de la partie gauche de la bande médiane le point de la partie droite de la bande médiane le plus proche.

Nb : Les tris sont en $n \log n$. Le point 7 de l'algorithme est linéaire car dans un rectangle de hauteur 2δ et de largeur δ qui contient plus de 7 points il y a deux points a une distance $< \delta$.